Autenticidade na aprendizagem baseada em projetos para desenvolvimento de software: Uma Revisão Sistemática de Literatura

Vinícius Gomes Ferreira[1,2], **Edna Dias Canedo**[2]

[1]Instituto Federal de Educação, Ciência e Tecnologia de Goiás-(IFG), Formosa–GO, Brasil

[2]Departamento de Ciência da Computação, Universidade de Brasília (UnB), Brasília–DF, Brasil

vinicius.gomes@ifg.edu.br, ednacanedo@unb.br

Abstract. *This paper presents a systematic literature review (SLR) aimed at investigating how Project-Based Learning (PBL) can contribute to making software authentic. As a result of the RSL, it was identified that the Scrum is the most widely used software development process in PBL units that generate authentic software. The identified papers also report that students are usually divided into groups of 2 to 5 people during activities and that monitors plays an important role in the quality of the software produced. These results allow professors to establish guidelines that can be used in the formation of their PBL units for software development that aims at authenticity of the produced artifact.*

Resumo. *Este trabalho apresenta uma revisão sistemática de literatura (RSL) com o objetivo de investigar como a Aprendizagem Baseada em Projetos (ABP) pode contribuir para tornar um software autêntico. Como resultado da RSL, identificou-se que o Scrum é o processo de desenvolvimento de software mais utilizado em unidades de ABP que geram softwares autênticos. Os trabalhos identificados também relatam que os alunos são normalmente divididos em grupos de 02 a 05 pessoas durante as atividades e os monitores exercem um papel importante na qualidade dos softwares produzidos. Estes resultados permitem aos docentes estabelecer diretrizes que podem ser utilizadas na formação de suas unidades de ABP para desenvolvimento de software que visem a autenticidade do artefato produzido.*

1. Introdução

A Aprendizagem Baseada em Projetos (ABP) é uma técnica bastante utilizada como prática de ensino [Maćias 2012, Jazayeri 2015]. A ideia por trás da ABP está na possibilidade do aluno desenvolver seu conhecimento por meio de construção de artefatos que possam ser percebidos e avaliados por outros [Papert and Harel 1991], não mais aguardando que o professor lhe preencha com os saberes da humanidade, mas agindo ativamente em busca de informações, testando hipóteses, criando artefatos e lidando com problemas do mundo real, que são comumente relacionados à autenticidade da unidade pedagógica. A ABP possui características especiais que a tornam alinhada com as abordagens progressistas de educação muito difundidas atualmente. Entre elas, constata-se o apelo pela autonomia do aluno, pela construção de artefatos que possam ser percebidos por outras pessoas e pela relação de proximidade com características do mundo real, o que é o que se define como autenticidade na ABP [Condliffe et al. 2017].

DOI: 10.5753/cbie.sbie.2019.41

Uma unidade letiva autêntica é aquela, que além de ser muito próxima de situações que acontecem no mundo real, também envolve algum nível de qualidade do artefato produzido em sala de aula [Helle et al. 2006, McDermott et al. 2017]. No entanto, embora o conceito de autenticidade seja de interesse da psicologia e da educação, ele ainda é um conceito difícil de se definir e até controverso [McDermott et al. 2017], o que torna complicado, para os docentes e outros interessados em aplicações de ABP para desenvolvimento de software, conceber o delineamento de uma unidade letiva que aumente a chance de tornar o software autêntico.

Embora as atividades na ABP, no geral, sejam bem próximas das que os alunos encontrarão em sua vida laboral e alguns resultados de sua aplicação reportados na literatura sejam animadores [Thomas 2000, Condliffe et al. 2017], poucos trabalhos preocupam-se com a questão da autenticidade aplicada ao artefato e à utilidade que ele pode prover a um usuário ou comunidade de usuários que demandam seu desenvolvimento para o grupo participante da unidade de ABP. Um software útil desenvolvido em unidades de ABP possui benefícios tanto para o usuário demandante, que pode receber um software a custo zero, quanto para os próprios alunos, que tendem se motivar mais por causa do grau de realismo do projeto [Blumenfeld et al. 1991] e podem começar a construir portfólio desde seu período de graduação diminuindo, assim, o *gap* comum da falta de experiência na procura pelo primeiro emprego.

Este trabalho apresenta uma Revisão Sistemática da Literatura (RSL) sobre a Aprendizagem Baseada em Projetos (ABP), com o propósito de identificar como a autenticidade tem sido alcançada nas unidades de ABP que envolvem a construção de software. Os resultados reportados nessa RSL podem prover informações úteis à construção de um modelo de ABP para desenvolvimento de software que seja intencional na construção de artefatos autênticos, o que pode trazer benefícios de uso para seus demandantes.

2. Aprendizagem Baseada em Projetos

As abordagens de ABP são balizadas por *práticas de investigação* feitas pelos estudantes, por meio de *colaboração* entre eles e pelo *uso de tecnologias novas* para que seja possível responder *questões motrizes* [Marx et al. 1997]. A partir disso, os alunos criarão artefatos autênticos que representem o entendimento deles [Marx et al. 1997]. Embora haja inúmeras versões de ABP (ou de abordagens que poderiam ser chamadas de ABP), esses são pilares chaves de unidades de ABP já registrados pela literatura, onde há mais consenso [Condliffe et al. 2017]. A seguir é apresentada a definição de cada um desses pilares. **1. Questão motriz:** Uma questão direcionadora guia os projetos educacionais da ABP e a transforma em uma abordagem particularmente diferente de outras abordagens educacionais [Blumenfeld et al. 1991]. **2. Investigações:** Um processo de investigação é estimulado pela questão motriz e fornece respostas para ela em um movimento de retroalimentação [Marx et al. 1997]. **3. Ferramentas tecnológicas:** A tecnologia web é, na opinião de muitos estudiosos, uma ferramenta chave na ABP para apoiar os alunos na condução da investigação e também na produção dos artefatos [Blumenfeld et al. 1991, Marx et al. 1997]. **4. Colaboração:** Fazer projetos, conduzir investigações e também aprender é uma atividade social. Tomando como base a ideia de que um dos grandes desafios que os alunos enfrentarão na vida profissional é lidar com grupos e realizar trabalhos dentro deles, a ABP permite que os alunos pratiquem o trabalho em equipe [Blumenfeld et al. 1991].

Embora o conceito de autenticidade seja de interesse da psicologia e da educação, ainda é um conceito difícil de se definir e até controverso [McDermott et al. 2017]. A construção de conhecimento, a investigação e uma similaridade com o que acontece no mundo fora da escola são três parâmetros passíveis de serem usados para a classificação da autenticidade [Newmann and Archbald 1992]. De acordo com a filosofia construtivista, a pedagogia autêntica afirma que os alunos devem utilizar seu conhecimento prévio para se envolver com problemas, tarefas e desafios que se conectam com o mundo além da sala de aula. Isso significa que a aprendizagem autêntica parte de problemas autênticos, sendo um problema autêntico aquele que está fundamentado em um contexto do mundo real [Blumenfeld et al. 1991, Helle et al. 2006]. Como a ABP é um modelo construtivista, a autenticidade exerce nela uma grande importância. É por meio da autenticidade que se argumenta que é necessário que os alunos dominem as técnicas e ferramentas dos profissionais da área para o qual estão se preparando, desde a realização das atividades até a construção de objetos de conhecimento [McDermott et al. 2017]. Os projetos desenvolvidos em uma unidade de ABP incorporam várias características que são responsáveis pela definição de autenticidade. Estas características podem incluir o tema, as tarefas, os papéis que os alunos desempenham, o contexto em que o trabalho do projeto é realizado e os colaboradores que trabalham com os alunos no projeto [Thomas 2000].

Uma das práticas da ABP que reforça o entendimento de que o que é produzido pelos alunos deve transcender a utilidade meramente didática, é o fato de que os artefatos construídos em unidades de ABP devem ser submetidos à crítica externa [Blumenfeld et al. 1991, McDermott et al. 2017]. Quando a transferência do sucesso de uma unidade autêntica de aprendizagem se traduz na disponibilização dos artefatos para uso da comunidade externa, essa comunidade se beneficia da aplicação da ABP ao receber um objeto útil e valoroso para uso e mitigação de seus problemas. A ideia de que o artefato gerado pela unidade de ABP pode resolver problemas reais causa efeitos positivos na motivação dos alunos, que é um dos elementos que, se não forem bem trabalhados durante a execução dos projetos, pode levá-los ao fracasso [Blumenfeld et al. 1991]. E, dessa maneira, a forma de se deve criar condições nas quais as questões que os alunos buscam e os artefatos que eles criam não sejam do tipo de ”trabalhos de escola”é algo que merece ser explorado [Lepper 1988].

3. Revisão Sistemática da Literatura

A Revisão Sistemática da Literatura (RSL) consiste em uma sequência de passos bem definidos para mapear, encontrar, avaliar criticamente, consolidar e agregar resultados de estudos primários relevantes sobre uma questão de pesquisa específica [Kitchenham 2004]. As fases que compõem seu modelo de trabalho são: Planejamento, Condução e Publicação dos Resultados [Kitchenham 2004]. Tendo em mente que é importante conhecer os aspectos técnicos do processo de construção de software nas unidades de ABP ao mesmo tempo em que se preza pelos benefícios pedagógicos já atribuídos à autenticidade, foram construídas questões de pesquisa (QP) que abordam tanto aspectos técnicos quanto pedagógicos das intervenções estudadas. As seguintes questões de pesquisa (QP) foram definidas: **QP.1.** Como o conteúdo teórico e técnico é ensinado aos alunos enquanto eles desenvolvem os projetos propostos nas disciplinas? **QP.2.** Qual o tempo disponível utilizado nas aplicações de aprendizagem baseada em projetos para criação de softwares que serão ou podem ser usados em um contexto real? **QP.3.** Quais modelos e/ou técnicas

de engenharia de software são usados no projeto? **QP.4.** Como são formados os grupos de alunos que participam do desenvolvimento de software em sala de aula?

Como estratégia de pesquisa, foi utilizada a Busca Automática nas bases de dados eletrônicas Scopus, Web of Science, ACM Digital Library e IEEE Xplore por intermédio da *string* de busca: **"software development"AND (classroom OR "learning"OR "teaching"OR education*) AND ("project-based learning"OR PBL OR "project based learning"OR "ProjBL"OR "PjBL")**. Também foi utilizada a Busca Manual em anais de revistas/jornais e conferências com reconhecida relevância na área de informática na educação, a saber: IEEE Frontiers in Education Conference, ACM Technical Symposium on Computer Science Education, Conference on Innovation and Technology in Computer Science Education, International Conference on Software Engineering. No total foram identificados **485 trabalhos**. Na primeira etapa da RSL, os artigos foram aceitos e/ou rejeitados com base na leitura do resumo e palavras chaves. Nesta etapa foram selecionados **179 artigos**.

Na segunda etapa, os estudos foram selecionados e/ou rejeitados a partir da leitura completa do artigo e confronto com critérios de inclusão e exclusão. Os critérios de inclusão foram: (1) O trabalho deve usar a abordagem de aprendizagem baseada em projetos; (2) O trabalho deve abordar o desenvolvimento de software autêntico; (3) O trabalho deve apresentar a entrega dos artefatos produzidos durante o desenvolvimento do software; (4) O modelo deve ter sido validado utilizando um estudo de caso prático e algum usuário deve ter avaliado positivamente o software ou o software deve estar sendo usado em alguma organização. Os critérios de exclusão foram: (1) Trabalhos fora do escopo de aprendizagem baseada em projetos; (2) O modelo proposto não foi validado utilizando um estudo de caso prático; (3) O software produzido por meio da abordagem não é funcional; (3) Não houve aprendizagem baseada em projetos aplicada para educação em Ciência da Computação; (4) O artefato desenvolvido pela abordagem não está fundamentada em um contexto do mundo real; (5) O trabalho não apresenta um estudo empírico; (6) Não foi possível ter acesso ao artigo completo de maneira gratuita através das bases de dados; e os artigos nos quais foram identificado a ocorrência de qualquer um deles foram rejeitados. No final, foram selecionados **38 artigos**. Um segundo pesquisador efetuou a leitura completa dos estudos selecionados e aplicou os critérios de qualidade adotados. Após essa fase, **23 estudos primários** foram selecionados para responder (fase de extração de dados) as questões de pesquisa que foram propostas.

3.1. Resultados da RSL

3.1.1. QP.1

O modelo de ABP sempre deve levar em consideração o aprendizado dos alunos que dela fazem parte [Blumenfeld et al. 1991, Helle et al. 2006]. É importante salientar que, em muitas aplicações de ABP, os alunos participantes estão em um nível mais avançado de conhecimento, depois de terem estudado os conceitos fundamentais da computação e engenharia de software em outras disciplinas [Cadenas et al. 2014, Schefer-Wenzl and Miladinovic 2017, Marques et al. 2018]. Entretanto, mesmo nesses casos, é necessário algum tipo de ensino formal aos alunos envolvidos nas unidades de ABP e esse ensino nem sempre será possível ser executado em sala de aula. Por causa disso, em algumas aplicações de ABP há a escalação de mais de

um professor [Cadenas et al. 2014] e até de uma equipe de monitores não docentes para auxiliar na tarefa de ensino dos alunos [Abler et al. 2011, Kizaki et al. 2014, Aibin and Hunter 2018] e acompanhá-los no desenvolvimento de seus projetos. Isso auxilia na manutenção da qualidade dos artefatos finais destes projetos, tornando-os mais autênticos [Marques et al. 2018].

O trabalho apresentado por Marques et al. [Marques et al. 2018] mostra evidências de que os alunos sob supervisão tiveram valores significantemente melhores do que os alunos não supervisionados em termos de efetividade, coordenação e senso de pertencimento a um time. Já Aibin e Hunter [Aibin and Hunter 2018] concluíram que não há diferenças significativas entre os dois grupos em relação ao *feedback* do cliente (o grupo não supervisionado é apenas um pouco melhor) e à satisfação dos alunos. A forma de distribuição dos conteúdos em uma ABP pode acontecer de inúmeras formas, entre elas, Instrução Direta [Cadenas et al. 2014, Mahnič 2015, Mahnič 2017, Yuen 2015, Olszewska et al. 2017], *Coaching* [Detmer et al. 2010, Intayoad 2014, Cadenas et al. 2014, Olszewska et al. 2017], disponibilização de materiais de ensino para consumo fora de sala de aula [Cadenas et al. 2014, Detmer et al. 2010, Martinez-Arias and Sarria M. 2013, Abler et al. 2011, Aibin and Hunter 2018], alunos ensinando outros alunos [Fagerholm et al. 2018, Kilamo et al. 2012, Abler et al. 2011], aulas práticas ou de laboratório [Llopis and Guerrero 2018, Martinez-Arias and Sarria M. 2013, Intayoad 2014] e ensino sob demanda [Schefer-Wenzl and Miladinovic 2017].

3.2. QP.2

O tempo pode se tornar um fator muito limitante em uma unidade de ABP. Não apenas em relação à entrega dos conteúdos, mas também em relação à qualidade geral do artefato desenvolvido. As restrições naturais de um ambiente de ensino depõem contra muitos princípios das práticas ágeis, por exemplo. Em um ambiente ágil, é esperado que as equipes trabalhem juntas a todo momento, exercitem a reflexão sobre o seu trabalho todos os dias e, se possível, mantenham o cliente sempre por perto [Kizaki et al. 2014]. As abordagens de ABP são normalmente executadas dentro de unidades letivas de um semestre [Aibin and Hunter 2018, Marques et al. 2018, Llopis and Guerrero 2018, Mahnič 2017, Mahnič 2015, Pariata and Montaño 2014, Rick et al. 2012, Kilamo et al. 2012, Schefer-Wenzl and Miladinovic 2017, Intayoad 2014], embora essa não seja uma regra geral. Existem casos em que a unidade de ABP é executada em um tempo inferior a um semestre [Cadenas et al. 2014, Yuen 2015], e outras em que ela dura mais de um semestre letivo [Ilkan et al. 2010, Kizaki et al. 2014, Foster et al. 2018, Martinez-Arias and Sarria M. 2013, Abler et al. 2011]. Embora a maioria dos estudos encontrados na literatura estejam dentro de um semestre letivo, a quantidade de semanas e horas disponíveis varia bastante.

Intayoad [Intayoad 2014] descreve uma distribuição de horas diferente para três disciplinas e uma unidade de ABP, uma vez que sua proposta é interdisciplinar. Nela participam os alunos de uma disciplina de Programação Orientada a Objetos, Interação Humano Computador e Tópicos Selecionados I. As horas em sala de aula podem ser usadas tanto para aulas teóricas, quanto para construção do projeto ou realização de atividades de laboratório. Na abordagem apresentada por Kilamo et al. [Kilamo et al. 2012], as horas em sala de aula são bem definidas, sendo a primeira delas para discutir algum

tópico específico do conteúdo teórico da disciplina e a última para discussão relacionada às contribuições feitas pelos alunos ao projeto durante a semana. Aibin e Hunter [Aibin and Hunter 2018] também descrevem uma possível divisão de tempo que pode ser feita para uma abordagem ABP, embora nem todas as horas sejam dedicadas às reuniões de sala de aula. Neste trabalho, é mencionado duas aplicações de ABP que possuem tempo reservado para reunião do time de alunos, encontros de sala de aula e encontro com o cliente. O que marca a diferença entre as duas aplicações é o tempo reservado para encontros com o tutor. Uma forma diferente de utilizar o tempo disponível é apresentada no trabalho de Ilkan et al. [Ilkan et al. 2010]. Os autores desenham todo o currículo do qual a unidade de ABP faz parte para um período de dois semestres e um treinamento de 40 dias dedicados à ABP. No treinamento, os alunos passam um tempo na indústria em busca de um problema que deve ser definido/resolvido no tempo proposto. Já no primeiro semestre, eles marcam reuniões semanais com seu orientador, e no semestre seguinte desenvolvem a solução em uma disciplina específica.

3.3. QP.3

Um modelo de ABP possui alguns princípios, mas em essência é genérico para que possa comportar unidades letivas de qualquer disciplina [Condliffe et al. 2017]. Embora a produção de software dentro de uma ABP deva seguir os mesmos princípios de um ABP convencional, frequentemente ela é conduzida utilizando um processo de desenvolvimento de software e técnicas de engenharia de software. Um modelo bastante comum para instanciações de ABP é o *Scrum* [Rupakheti et al. 2018, Aibin and Hunter 2018, Llopis and Guerrero 2018, Mahnič 2015, Mahnič 2017, Olszewska et al. 2017, Yuen 2015, Kizaki et al. 2014, Fagerholm et al. 2018]. O *Scrum* é capaz de prover controle sobre o progresso do projeto e de garantir um ritmo constante de desenvolvimento. Sua configuração nem sempre é a mesma em todas as aplicações, havendo divergências no tamanho da *Sprint* (de 1 a 4 semanas) [Olszewska et al. 2017, Yuen 2015, Mahnič 2015, Mahnič 2015, Mahnič 2017, Rupakheti et al. 2018, Fagerholm et al. 2018], e na forma como os eventos *Sprint Planning*, *Daily Scrum* (este sendo aplicado semanalmente e não diariamente), *Sprint Review* e *Sprint Retrospective* são conduzidos [Olszewska et al. 2017, Yuen 2015, Fagerholm et al. 2018, Mahnič 2017, Kizaki et al. 2014]. Embora Martinez-Arias e Gerardo [Martinez-Arias and Sarria M. 2013] não usem o *Scrum* no maior projeto de sua unidade ABP, faz uso dele em conjunto com o *eXtremme Programming* (XP) para ensinar métodos ágeis aos alunos por meio de pequenos projetos que também são autênticos.

Algumas técnicas isoladas do XP também podem ser utilizadas dentro de uma unidade de ABP para viabilizar tanto o aprendizado quanto a construção de um software. Os autores dos estudos primários identificados nessa RSL fazem uso de integração contínua [Rupakheti et al. 2018, Mahnič 2015, Kilamo et al. 2012] e *test-driven development* [Schefer-Wenzl and Miladinovic 2017]. Também existem relatos do uso de *ticket driven development* [Kizaki et al. 2014], acompanhamento de *bugs* [Kilamo et al. 2012, Olszewska et al. 2017, Kizaki et al. 2014], rastreamento de trabalho executado [Llopis and Guerrero 2018, Detmer et al. 2010, Schefer-Wenzl and Miladinovic 2017, Abler et al. 2011, Marques et al. 2018], e ferramentas de ”desenvolvimento puxado”como o *Kanban* [Mahnič 2015, Fagerholm et al. 2018, Rupakheti et al. 2018]. Embora os métodos ágeis sejam populares em unidades de ABP para desenvolvimento

de software, Marques et al. [Marques et al. 2018] argumentam que o uso de métodos prescritivos seja mais recomendados para alunos inexperientes e faz a aplicação de um modelo chamado *Simple Software Process*. Em seu modelo de Fábrica de Software, Pariatra e Montaño [Pariata and Montaño 2014] propõem o uso do modelo Cascata e sugere a divisão dos alunos em grupos especializados (Desenvolvimento, Documentação, Projeto e Teste). Além dele, Intayoad [Intayoad 2014] também faz uso de um método prescritivo, providenciando a divisão de alunos em tarefas especializadas de acordo com a disciplina que cursam e Martinez-Arias e Gerardo [Martinez-Arias and Sarria M. 2013] declaram o uso do RUP (*Rational Unified Process*) em sua abordagem. Finalmente, em alguns casos, processos criativos podem ser usados para emergir ideias inovadoras dos participantes da ABP, entre os quais é possível citar os *hackathons* [Schefer-Wenzl and Miladinovic 2017], *brainstorming* e *Design Thinking* [Pham et al. 2018]. Como também, há evidências do uso de estratégias de desenvolvimento distribuído [Rupakheti et al. 2018, Kizaki et al. 2014, Cadenas et al. 2014, Olszewska et al. 2017, Kilamo et al. 2012], justificada pela dificuldade em se manter os alunos todos os dias trabalhando juntos [Kizaki et al. 2014].

3.4. QP.4

Um aspecto importante das unidades de ABP é o fato de que elas auxiliam o desenvolvimento de habilidades não técnicas relevantes para a atuação na indústria de software que só podem ser construídas por intermédio da aprendizagem social. No entanto, os professores dessas unidades podem enfrentar problemas com turmas com uma grande quantidade de alunos, visto que recomenda-se adotar a formação de pequenos grupos [Llopis and Guerrero 2018], por causa dos problemas que isso pode causar à comunicação e à própria coordenação do trabalho. A maioria dos trabalhos selecionados na RSL dividem os alunos em grupos de 2 a 5 pessoas [Ilkan et al. 2010, Cadenas et al. 2014, Foster et al. 2018, Aibin and Hunter 2018, Detmer et al. 2010, Intayoad 2014, Pham et al. 2018, Pariata and Montaño 2014, Olszewska et al. 2017, Mahnič 2017, Yuen 2015], cada qual fazendo sua própria versão do projeto. Em alguns casos, esses projetos vêm de uma mesma coleção de requisitos [Llopis and Guerrero 2018, Cadenas et al. 2014, Foster et al. 2018, Pariata and Montaño 2014, Rupakheti et al. 2018], podendo ter como estratégia a divisão dos grupos para tratar de um mesmo projeto, com cada aluno trabalhando em um conjunto de recursos do artefato [Rupakheti et al. 2018] ou em aspecto transversal deste, como documentação, *design*, desenvolvimento e teste [Intayoad 2014]. Em outros são projetos diferentes para clientes diferentes [Ilkan et al. 2010, Aibin and Hunter 2018, Detmer et al. 2010, Intayoad 2014, Pham et al. 2018, Olszewska et al. 2017, Yuen 2015], para os quais há a possibilidade de adição de um elemento de competição como forma de motivação aos alunos e de obtenção de melhores resultados nos projetos [Llopis and Guerrero 2018].

Foster et al. [Foster et al. 2018] difere dos demais trabalhos apresentados no sentido de que vários grupos de alunos foram se revezando ao longo do tempo em um mesmo projeto, obrigando-os a terem que gerenciar o conhecimento de um grupo para o outro. No caso do trabalho de Cadenas et al. [Cadenas et al. 2014], os primeiros projetos alimentaram os projetos posteriores e também tiveram os desafios da gestão do conhecimento entre os grupos de projetos ao longo do tempo. Rupakheti et al. [Rupakheti et al. 2018]

apresentam uma maneira não usual de se formar os times, permitindo que os grupos façam propostas de soluções de software para serem desenvolvidos e tentem convencer os demais colegas a trabalhar com eles em suas propostas. É uma abordagem com certa similaridade ao apresentado por Detmer et al. [Detmer et al. 2010] na aplicação de ABP para uma disciplina de Linguagem de Programação em Java, onde os alunos selecionados como líderes indicaram com quem gostariam trabalhar e com quem não gostariam de trabalhar.

4. Conclusão

Levando em consideração as respostas encontradas para as questões de pesquisa que foram definidas neste trabalho, é possível afirmar que tornar um software autêntico em uma unidade de ABP envolve múltiplos fatores, dentro os quais é possível encontrar o suporte educacional aos alunos, a montagem adequada de um processo de engenharia de software e o tempo disponível para a execução de uma unidade de ABP. Além disso, foi possível identificar que o *Scrum* destaca-se como o método com mais aplicações, levando a acreditar que sua adoção pode ajudar na construção da autenticidade do artefato.

Embora alguns autores argumentem a favor de métodos prescritivos para alunos não experientes com o desenvolvimento de software, a maioria deles descreve ferramentas específicas relacionadas aos modelos ágeis para obtenção de melhores resultados na condução da ABP e no desenvolvimento do artefato. Muitos trabalhos coletados nessa RSL apontaram a importância de se fazer uma divisão dos grupos de alunos durante as atividades de construção de software em uma unidade de ABP. Além disso, permitiu identificar que a competição entre os alunos apresenta benefícios úteis para o alcance de um artefato autêntico. A direção dos trabalhos futuros pode estar na construção de um modelo de ABP para desenvolvimento de software que seja intencional na construção de artefatos autênticos, no sentido da utilidade que eles proveem para seus demandantes.

Referências

Abler, R., Coyle, E., Kiopa, A., and Melkers, J. (2011). Team-based software/system development in a vertically-integrated project-based course. *Proceedings - Frontiers in Education Conference, FIE*, pages T3F–1–T3F–7.

Aibin, M. and Hunter, A. (2018). On Faculty Supervision in Industry Projects. *Western Canadian Conference on Computing Education*, pages 1–5.

Blumenfeld, P. C., Soloway, E., Marx, R. W., Krajcik, J. S., Guzdial, M., and Palincsar, A. (1991). Motivating project-based learning: Sustaining the doing, supporting the learning. *Educational psychologist*, 26(3-4):369–398.

Cadenas, J. T., Rodríguez, R., and Omaña, M. (2014). Conceptualizing a didactics experience: Mini-Project Software Development. *Proceedings of the 2014 Latin American Computing Conference, CLEI 2014*.

Condliffe, B., Quint, J., Visher, M. G., Bangser, M. R., Drohojowska, S., Saco, L., and Nelson, E. (2017). Project-Based Learning A Literature Review Working Paper Prepublication copy: Release. Working paper.

Detmer, R., Li, C., Dong, Z., and Hankins, J. (2010). Incorporating Real-world Projects in Teaching Computer Science Courses. *Proceedings of the 48th Annual Southeast Regional Conference*, pages 24:1—-24:6.

Fagerholm, F., Hellas, A., Luukkainen, M., Kyllönen, K., Yaman, S., and Mäenpää, H. (2018). Designing and implementing an environment for software start-up education: Patterns and anti-patterns. *Journal of Systems and Software*, 146:1–13.

Foster, D., Gilardi, F., Martin, P., Song, W., Towey, D., and White, A. (2018). Students as co-producers in a multidisciplinary software engineering project: addressing cultural distance and cross-cohort handover. *Teachers and Teaching: Theory and Practice*, 24(7):840–853.

Helle, L., Tynjälä, P., and Olkinuora, E. (2006). Project-Based Learning in Post-Secondary Education – Theory, Practice and Rubber Sling Shots. *Higher Education*, 51(2):287–314.

Ilkan, M., Amca, H., and Iscioglu, E. (2010). Grooming IT Students for Industry Through Industrial Training and Graduation Project Work. *Information-an International Interdisciplinary Journal*, 13(4):1219–1242.

Intayoad, W. (2014). PBL framework for enhancing software development skills: An empirical study for information technology students. *Wireless Personal Communications*, 76(3):419–433.

Jazayeri, M. (2015). Combining Mastery Learning with Project-Based Learning in a First Programming Course: An Experience Report. *Proceedings - International Conference on Software Engineering*, 2:315–318.

Kilamo, T., Hammouda, I., and Chatti, M. A. (2012). Teaching collaborative software development: A case study. *Proceedings - International Conference on Software Engineering*, pages 1165–1174.

Kitchenham, B. (2004). Procedures for performing systematic reviews. *Keele, UK, Keele University*, 33(TR/SE-0401):28.

Kizaki, S., Tahara, Y., and Ohsuga, A. (2014). Software development PBL focusing on communication using scrum. *Proceedings - 2014 IIAI 3rd International Conference on Advanced Applied Informatics, IIAI-AAI 2014*, pages 662–669.

Lepper, M. R. (1988). Motivational Considerations in the Study of Instruction. *Cognition and Instruction*, 5(4):289–309.

Llopis, F. and Guerrero, F. G. (2018). Introducing competitiveness and industry involvement as learning tools. *IEEE Global Engineering Education Conference, EDUCON*, 2018-April:298–307.

Maćias, J. A. (2012). Enhancing project-based learning in software engineering lab teaching through an e-portfolio approach. *IEEE Transactions on Education*, 55(4):502–507.

Mahnič, V. (2015). The capstone course as a means for teaching agile software development through project-based learning. *World Transactions on Engineering and Technology Education*, 13(3):225–230.

Mahnič, V. (2017). Student projects as a means of cooperation between academia and industry: Some experiences in the area of software engineering education. *World Transactions on Engineering and Technology Education*, 15(3):239–244.

Marques, M., Ochoa, S. F., Bastarrica, M. C., and Gutierrez, F. J. (2018). Enhancing the Student Learning Experience in Software Engineering Project Courses. *IEEE Transactions on Education*, 61(1):63–73.

Martinez-Arias, J. C. and Sarria M., G. M. (2013). Didactic and interdisciplinary experiences in a software engineering course. *Proceedings - Frontiers in Education Conference, FIE*, pages 1800–1805.

Marx, R. W., Blumenfeld, P. C., Krajcik, J. S., and Soloway, E. (1997). Enacting Project-Based Science. *The Elementary School Journal*, 97(4):341–358.

McDermott, R., Zarb, M., Daniels, M., Nylén, A., Pears, A., Isomöttönen, V., and Caspersen, M. (2017). The authenticity of 'Authentic' assessment some faculty perceptions. *Proceedings - Frontiers in Education Conference, FIE*, 2017-Octob:1–9.

Newmann, F. M. and Archbald, D. A. (1992). The nature of authentic academic achievement. *Toward a new science of educational testing and assessment*, pages 71–83.

Olszewska, M., Ostroumov, S., and Olszewski, M. (2017). To agile or not to agile students (with a twist): Experience report from a student project course. *Proceedings - 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2017*, pages 83–87.

Papert, S. and Harel, I. (1991). Situating constructionism. *Constructionism*, pages 1–14.

Pariata, M. and Montaño, N. (2014). Software Factory, from professional environment to academic environment: Proposal to build competences through authentic activities in the context of software engineering. *Proceedings of the 2014 Latin American Computing Conference, CLEI 2014*, pages 1–10.

Pham, Y. D., Fucci, D., and Maalej, W. (2018). A First Implementation of a Design Thinking Workshop During a Mobile App Development Project Course. *ACM/IEEE International Workshop on Software Engineering Education for Millennials*, pages 56 –63.

Rick, D., Morisse, M., and Schirmer, I. (2012). Bringing contexts into the classroom. *Proceedings of the 7th Workshop in Primary and Secondary Computing Education on - WiPSCE '12*, page 105.

Rupakheti, C. R., Hays, M., Mohan, S., Chenoweth, S., and Stouder, A. (2018). On a pursuit for perfecting an undergraduate requirements engineering course. *Journal of Systems and Software*, 144:366–381.

Schefer-Wenzl, S. and Miladinovic, I. (2017). Game Changing Mobile Learning Based Method Mix for Teaching Software Development. *Proceedings of the 16th World Conference on Mobile and Contextual Learning - mLearn 2017*, pages 1–7.

Thomas, J. W. (2000). A Review of Research on Project-Based Leaming. Working paper.

Yuen, T. T. (2015). Scrumming with educators: Cross-departmental collaboration for a summer software engineering capstone. *Proceedings - 2015 International Conference on Learning and Teaching in Computing and Engineering, LaTiCE 2015*, pages 124–127.